Thoughts on How to Distribute the Bare Minimum for Analysis

Natalia Ratnikova,
Fermilab

16 May, 2006

Software Development Tools Meeting.

Use Case

- Physicist wants to run root-based analysis on his/her laptop without installing full CMSSW development environment.
- Goal:
  - Let users easily install a light version of CMSSW software containing data classes and corresponding dictionaries on their laptops
- Examples: FWLite based tutorials at:

https://twiki.cern.ch/twiki/bin/view/CMS/May06CPTweekTutorials

General Requirements

- Package includes a bare minimum for analysis
- Package is built as a subset of the CMSSW release:
  - Based on the same configuration version (SCRAMToolBox tag), and selects a subset of required external tools (CMSSW/config/requirements).
  - Based on the same source code
  - Package is considered a part of the release provided functionality and thus undergoes all standard release procedures: nightlies, integration, validation, release notes, packaging, etc…
  - Downloaded/installed via standard CMSSW tool
- More ???

Possible Solutions

Distribution with build capabilities

OR

Runtime based distribution

???

Distribution With Build Capabilities

- Identify a bare minimum subset of external tools (using <select> tag in config/requirements)
- Identify a bare minimum list of subsystems to build, e. g.: DataFormats, SimDataFormats .
- Bootstrap/setup/build/test and create rpm as usual, but only for bare minimum.
- Review the *internal dependencies* (BuildFiles),e.g. if unwanted subsystems are pulled out during the build, and *external dependencies* (selected tools), e.g. if unresolved.
- Package/install using new project name but the global version : FWLite CMSSW_0_6_0

Runtime Based Distribution

- Package only “bare minimum” of environment that you get when executing scram runtime.
  - Runtime based distribution will be even more compact without unneeded sources, binaries and build tools
  - No external dependencies: all in one package; no scram
  - User only needs to source some setup.sh/csh script to be ready to run the application, or even have a wrapper around the root.exe put in $HOME/bin/fwlite, which can be executed directly.
- Runtime based package can be built from FWLite to be self contained by construction. Or from the full CMSSW release, with additional validity check(e.g. touch each object type).

Realization

- Prototyping would take a week or so
- Need to prioritize wrt other tasks